PC MAGAZINE

Automate Long Fills In Excel

November 19, 2003
By Mark J. Brickley, Alan C. Cutting

How many times have you dragged a formula down an Excel sheet only to find yourself a thousand rows past your destination? You can easily solve that frustration with a few lines of VBA code. If you've never written anything in Microsoft Visual Basic for Applications (VBA), this is a great opportunity to learn some of the basics.

Let's begin by asking Excel for a hand in writing the VBA code using Excel's recorder to create a macro. Once you have the basic code, you can modify it to your specific needs. Since our new Excel feature deals with copying formulas, we'll let the recorder watch us copy a selection of cells down a range.

Select a few cells, such as the range C5 through E6 (don't worry that the cells are empty). Now start the recorder by selecting *Tools* | *Macro* |*Record New Macro*. Enter *BigFill* as the name of the macro and, if you like, designate a shortcut key. Click on OK. Use the copy handle to grab the range of selected cells and drag it down a few rows, say to E9. Stop the recorder either by clicking on the Stop button on the floating recorder toolbar or by selecting *Tools* | *Macro* | *Stop Recording*.

Now press Alt-F11 and you should find yourself in the VBA editor. If your code isn't visible, you'll need to get to *Module1*. If you don't see a project window on your left, select *View* | *Project Explorer*. Then double-click on *Modules* and again on *Module1*. You should now see the code as in Figure 1.

Believe it or not, there are very few changes to make. The two locations that cite a specific range (C5:E9) need to be changed to match the fill area. We'll use an `InputBox` to ask for the number of rows to be filled and simply have Excel count the number of highlighted columns. We'll store the information we gather in the variables `numRows` and `numCols` for use later in the routine. And we'll define a third variable called `fillRange` to hold information about the range involved (more about that below).

`DIM` statements are used to define the kind of variables you want. `FillRange` needs to hold information about a range, while `numRows` and `numCols` hold whole numbers (integers).

Add the necessary lines of code—the first four lines under `Sub BigFill()` in the figure—just before the line that starts with `Selection.AutoFill`.

The `InputBox` function uses two arguments: `Prompt` (the question we are asking the user) and `Title` (the title of the InputBox). And notice the dots in the statement, which grab the number of columns (`numCols = Selection.Columns.Count`). In VBA, these are used to separate an object (such as a `Selection`) from any properties or methods it may have. One of the `properties` of a selected area in Excel is that it contains columns. A `method` we can apply to this area is to count these columns.

The next step is to define a range that is unknown at the time we write our routine. For this we need to know two things: where the cursor is and the address at the other end of the fill range. The cursor's location is fairly simple; VBA provides an object called the `ActiveCell`. We can find the end of the range using a property of the `ActiveCell`, called `Cells`.

The `Cells` property treats the row and column values as an index, so that the rows and columns get numbered starting from wherever the cursor is. So assuming the cursor (the `ActiveCell`) is in the first row and first column, the command `ActiveCell.Cells(2,3)` refers to a cell in the second row, third column.

You'll notice that the addressing scheme is *Row*, then *Column* (opposite of the way Excel references cells).

Since we know the number of rows and columns to be filled (`numRows` and `numCols`), those two values will define the ending location of our fill range. The range to be filled will extend from the cell in the upper-left-hand side of the range to the cell at the lower-right-hand corner of the range. The upper left cell will be `Cells(1,1)` and the other corner will depend on the number of rows and columns; it will be `Cells(numRows, numCols)`. So starting at the cell that is currently active (the first cell in the group), that range will go from `Cells(1,1)` to `Cells(numRows, numCols)`. Here's the VBA statement:

```
ActiveCell.Range(Cells(1,1), Cells(numRows,numCols) )
```

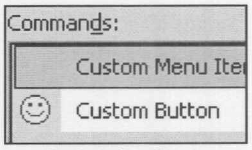
To complete the code, set the variable that will hold the range equal to the line above and substitute this range with our starting range `C5:E9`. Your completed code should look like the lines in Figure 1. Now return to your worksheet and run the macro using your shortcut key or by entering Alt-F8 to bring up the *Macro* dialog and choosing *Run*.

You can make the macro even more useful by adding it to Excel's *Tools* menu.

Adding BigFill to Excel's Tools Menu

Online Extra

The goal is to create a new item on Excel's Tools menu that lets you easily run the BigFill macro. Excel has a very simple method for attaching macros to any menu. Select Tools | Customize and choose the Commands tab. Scroll down the categories list until you reach Macros and use your mouse to drag Custom Menu Item up to the Tools menu at the top of your Excel sheet. Position the cursor at any level in the Tools menu and when you release the mouse, Custom Menu Item will land on the menu. Right-click on it to rename the new item and assign it to a macro. Replace the default name (Custom Menu Item) with Big&Fill and assign the macro to BigFill. (An ampersand in front of any letter turns that letter into a keyboard accelerator.) Now you can run your new fill macro using Alt-T (for the Tools menu), followed by F for Fill.



Customize Menu

ENLARGE ⊞

You can get this new menu to appear whenever you work with Excel on your own machine. The trick is to save your project under a special name—Personal.xls—and in a special place, the XLStart folder. The location of this folder varies. In Windows XP, the path is C:\Program Files\Microsoft Office\Office 10\XLStart. Normally, Personal.xls opens as a hidden file when you launch Excel. To ensure it stays hidden, select Windows | Hide. To save it, use the X in the upper-right-hand corner to exit Excel; you'll be prompted to save the file. Be sure to use the name Personal.xls and set the location to the XLStart folder. When you restart Excel, your workbook should be Book1 (not Personal) and it should have BigFill on the Tools menu.